Abstract

a unified lattice model for static analysis of programs by construction or approximation of fixpoints

Patrick Cousot and Radhia Cousot, 1977
Motivation (for static analysis)

Say you’ve written code that you *really* don’t want bugs in...

.....like the controls for some rocket boots.
Motivation

You want to reason about

Note: These sketches, and the intuition behind them, are from Patrick Cousot’s website!
Motivation

To make sure you’re safe
Motivation

....but you can’t analyze code perfectly
Motivation

Testing is dangerous...

Forbidden zone

Test of a few trajectories

Possible trajectories
Motivation

Luckily you have an ally…

STATIC ANALYSIS
Motivation

With the power of... Abstraction!

Better safe than sorry!
History – before this paper

• Early 70s work in data flow, type systems, etc
• As well as mathematical semantics
This paper

Uses mathematical semantics to give a grand unified theory of static analysis

Trivia:
Based on authors’ work in interval analysis
Initially a 100 page handwritten manuscript submitted to the 4th POPL
After this paper

• Rich literature on static analysis in just about any domain you want
• Further theoretical exploration of AI

• Future, more computer-aided design.
Some Definitions

• A **lattice** is a partial order \(< L, \leq \rangle\) such that every two elements have a unique supremum (join) and infimum (meet)

• A **complete lattice** has a unique join and meet for every non-empty subset of \(L\)

• A **semi-lattice** only has join (or meet)
Properties of $\alpha$, $\gamma$?

- Say we have a abstract, $c$ concrete corresponding
- Somehow want maps to pick the “best”
  - $\alpha(c) \leq a, \ c \leq \gamma(a)$
- Monotone
  - $p \leq q \Rightarrow \alpha(p) \leq \alpha(q)$
- $\forall x, x = \alpha(\gamma(x))$
- $\forall y, y \leq \gamma(\alpha(y))$
Examples of Abstractions

• Sets of Integers
• (unbounded) Intervals
• Congruence mod 2
• One value or Sign
YO DAWG, I HEARD YOU LIKE LATTICES

SO I PUT YOUR LATTICES IN A LATTICE
SO YOU CAN LATTICE WHILE YOU LATTICE
Interpretation

How do we actually use this?

Hi I'm a PL

Here have this semantics

Wow it’s great tnx
In this case

- Flowchart language
- Context-collecting semantics (cv)
- Local Interpretation Int(r,cv)
- Global Interpretation G-Int(cv)
- cv = G-Int(cv)
- Least fixed point
- Iterate G-Int(bot) to solve
Abstract Interpretation

An abstract interpretation \( I \) of a program \( P \) is a tuple
\[
I = \langle A-\text{Cont}, \circ, \leq, \top, \bot, \text{Int} \rangle
\]

\[
\{ \forall (a, x) \in \text{Arcs} \times A-\text{Cont}, \\
\gamma(\text{Int}(a, x)) \geq \text{Int}(a, \gamma(x)) \}
\]

and
\[
\{ \forall (a, x) \in \text{Arcs} \times C-\text{Cont}, \\
\text{Int}(a, \tilde{\alpha}(x)) \geq \alpha(\text{Int}(a, x)) \}
\]
Widening

- So, we’re done, right?
- No!
- We could be walking an infinite path

- Instead – jump! With over-approximations
9.5 Dual Approximation Methods

The lattice $\tilde{\text{A-Cont}}$ may be partitioned as follows:

![Diagram of lattice partitions](image)

- $X$ and $\tilde{\text{Int}}(X)$
- $X \preceq \tilde{\text{Int}}(X)$
- $X \succeq \tilde{\text{Int}}(X)$
- $X = \tilde{\text{Int}}(X)$

AKS $\rightarrow$ $\ellfp$ $\rightarrow$ gfp $\rightarrow$ DKS $\rightarrow$ $\tilde{\tau}$
Widening

Let $\widehat{\text{A-int}} : \widehat{\text{A-Cont}} \to \widehat{\text{A-Cont}}$ be such that:

9.1.1.1 $\{ \forall n \geq 0, C = \widehat{\text{A-int}}^n(\overline{1}) \text{ and } \text{not} (\widehat{\text{Int}}(C) \preceq C) \} \\ \implies \{ C \preceq \widehat{\text{Int}}(C) \preceq \widehat{\text{A-int}}(C) \}.$

9.1.1.2 Every infinite sequence $\overline{1}, \widehat{\text{A-int}}(\overline{1}), \ldots, \widehat{\text{A-int}}^n(\overline{1}), \ldots$ is not strictly increasing.

---

The binary operation $\lor$ called widening defined by:

9.1.3.1 $\lor : \text{A-Cont} \times \text{A-Cont} \to \text{A-Cont}$

9.1.3.2 $\forall (C, C') \in \text{A-Cont}^2, C \lor C' \leq C \lor C'$

9.1.3.3 Every infinite sequence $s_0, \ldots, s_n, \ldots$ of the form $s_0 = C_0, \ldots, s_n = s_{n-1} \lor C_n, \ldots$ (where $C_0, \ldots, C_n, \ldots$ are arbitrary abstract contexts) is not strictly increasing.

$\text{A-int} = \lambda(q, \overline{C_v}). \begin{cases} \overline{C_v(q)} \lor \overline{\text{Int}}(q, \overline{C_v}) & \text{if } q \in \text{W-arcs} \\ \text{else} \end{cases} \overline{\text{Int}}(q, \overline{C_v})$
Narrowing

• We might jump way too far
• Walk it back!

• Again, this may be an (infinitely) long walk

\[ s_m \geq \overline{\operatorname{Int}(s_m)} \geq \ldots \geq \overline{\operatorname{Int}^n(s_m)} \geq \ldots \geq \overline{C_v}. \]
Narrowing

9.3.4.1 $\Delta : \text{A-Cont} \times \text{A-Cont} \to \text{A-Cont}$

9.3.4.2 $\Psi(C, C') \in \text{A-Cont}^2,$  
$\{C \geq C'\} \implies \{C \geq C \Delta C' \geq C'\}$

9.3.4.3 Every infinite sequence $s_0, \ldots, s_n, \ldots$ of the form $s_0 = C_0, s_1 = s_0 \wedge C_1, \ldots, s_n = s_{n-1} \Delta C_n, \ldots$ for arbitrary abstract contexts $C_0, C_1, \ldots, C_n, \ldots$ is not strictly decreasing.

The approximated interpretation $D\text{-int} : \text{Arcs}^9 \times \text{A-Cont} \to \text{A-Cont}$ is defined by:

9.3.4.4 $D\text{-int} = \lambda(q, C_v)\cdot\begin{cases} C_v(q) \Delta \text{Int}(q, C_v) \\
\text{else} \quad \text{Int}(q, C_v) 
\end{cases}$

and

$D\text{-Int} = \lambda C_v . (\lambda q . D\text{-int}(q, C_v))$